LARGE SAMPLE SIMULATION OF FLICKER NOISE

James A. Barnes
Austron, Inc,
3300 Mitchell Ln. Suite 370
Boulder, CO 80302

Charles A. Greenhall
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109

ABSTRACT

The computer simulation of complex systems often requires efficient recursive algorithms to generate various noise types. Flicker noise often occurs in frequency and time systems and long samples (a million or more lags) are needed. In June of 1971 two papers on flicker noise simulation were published independently. Mandelbrot[1] based the simulation on the sum of several low pass filtered white, Gaussian noises. He showed that the noise level and the filter passband of each noise can be selected to provide an approximate flicker noise over a finite but arbitrary spectral range. The precision of fit to the flicker spectrum can be arbitrarily good depending on the number of independent elements used.

Barnes and Jarvis[2] based their simulation on a cascade of lead-lag filters. Similarly, the extent and goodness of fit of the output noise depends only on the number of filter stages used. The Barnes-Jarvis procedure has an exact inverse which has advantages over the Mandelbrot method in some statistical applications other than noise simulation. The Barnes-Jarvis method can be expressed as an ARIMA model, but certain problems arise from the loss of significant digits. The significant digits problem can be avoided by using the ARIMA model in "factored form".

Flicker noise is a noise whose power spectral density (PSD) varies approximately as the reciprocal of the Fourier frequency over a large, but finite range. Both methods cited can be used to simulate other power-law noises than just $1/f$ noise. Any noise with a PSD that varies as f^{α} for α in the range $-2 < \alpha < 0$ can be simulated directly by these methods. For an extension of the range, the output of these filters can be integrated or differentiated to obtain any power-law desired. The Barnes-Jarvis filter is also called a "constant argument" filter since the output phase (45 deg.) of the filter is a constant relative to the input phase for a sinusoidal input at any frequency. A pure integrator has a 90 degree lag.

THE BARNES-JARVIS FILTER

Barnes and Jarvis cascaded a series of lead-lag filters chosen to approximate a $1/f$ spectrum. The digital equivalent of a lead-lag filter can be expressed with a recursive filter[3] in the form:

$$Y_n = \phi Y_{n-1} + P_n - \phi P_{n-1} \tag{1}$$

where P_n is the input to the filter and Y_n is the output. The coefficients, ϕ and θ, determine the poles and zeros of the filter. The frequency $W = 2\pi f$ (W is used rather than ω to aid in comparison to computer program listings) at the knee in the transfer function (see Fig.1)

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
DEC 1987		**00-00-1987 to 00-00-1987**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Large Sample Simulation of Flicker Noise	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Austron, Inc,,3300 Mitchell Ln. Suite 370,Boulder,CO,80302**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Proceedings of the Nineteenth Annual Precise Time and Time Interval (PTTI) Applications and Planning Meeting, Redondo Beach, CA, 1-3 Dec 1987

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**15**	

determines the value of ϕ according to the relation:

$$\phi = 1 + \frac{\omega_0}{2}(\omega_0 - \sqrt{\omega_0^2 + 4}) \tag{2}$$

Conversely, the value of ϕ can be used to find the knee frequency according to the inverse of (2):

$$\omega_0 = (1 - \phi)/\sqrt{\phi} \tag{3}$$

The same functional form given in (2) and (3) also apply to the θ coefficient which forms the high frequency response of the lead-lag filter.

One next selects a ratio, R, $(R > 1)$ which determines the frequency ratios of successive filter stages. The closer that R is to unity the closer the filter is to an ideal $1/f$ spectrum; but, the more stages are required to fill a specific frequency range (see Fig.2). For flicker noise generation, the frequencies corresponding to successive knees (determined by the ϕ's) grow as R^2 as do the frequencies corresponding to the θ coefficients. In each stage, the frequency for the θ is R times the frequency for the ϕ.

The filter stage for the highest frequency filter is a special case. First, since we are dealing with real functions of finite length, the slope of the spectrum at the Nyquist frequency $(1/2T)$ is already zero of necessity. The corresponding θ, then, can be taken to be zero. The next problem is to determine a frequency corresponding to the first (highest frequency) ϕ. This selection is usually done by trial and error to obtain the best overall approximation to a $1/f$ spectrum (see Fig. 2 and Appendix A). The first ϕ is typically in the range of 0.3 to 0.5.

A simple program (in BASIC) to determine the ϕ's and θ's follows:

```
220    REM COMPUTE FACTORED PHI'S AND THETA'S
230    PH(1)=.45#:W=(1#-PH(1))/SQR(PH(1)):TH(1)=0
240    FOR N=2 TO M
250        W=W/R
260        TH(N)=1#+.5#*W*(W-SQR(W*W+4#))
270        W=W/R
280        PH(N)=1#+.5#*W*(W-SQR(W*W+4#))
290    NEXT N
```

where M is the number of lead-lag stages to be used, R is the frequency ratio, W is the angular frequencies of the corresponding ϕ's and θ's, and .45 is the initial ϕ selected from trial and error. The symbol "#" indicates double precision constants. All of the calculations should be carried out in double precision except for storing the final time series. The filter equations then become:

```
300    FOR N=1 TO NTOTAL
350        Y(1)=PH(1)*Y1(1)+P
360        FOR I=2 TO M
370            Y(I)=PH(I)*Y1(I)+Y(I-1)-TH(I)*Y1(I-1)
380            Y1(I-1)=Y(I-1)
390        NEXT I
400        Y1(M)=Y(M)
410    NEXT N
```

where NTOTAL is the total number of points to be generated. The inputs to the filter are the P-values, which are assumed to be random, normal deviates with zero mean and a variance determined by the particular simulation (unity, here). The filter output is the output of the M^{th}, or final stage: Y(M). Limited storage space may restrict how many

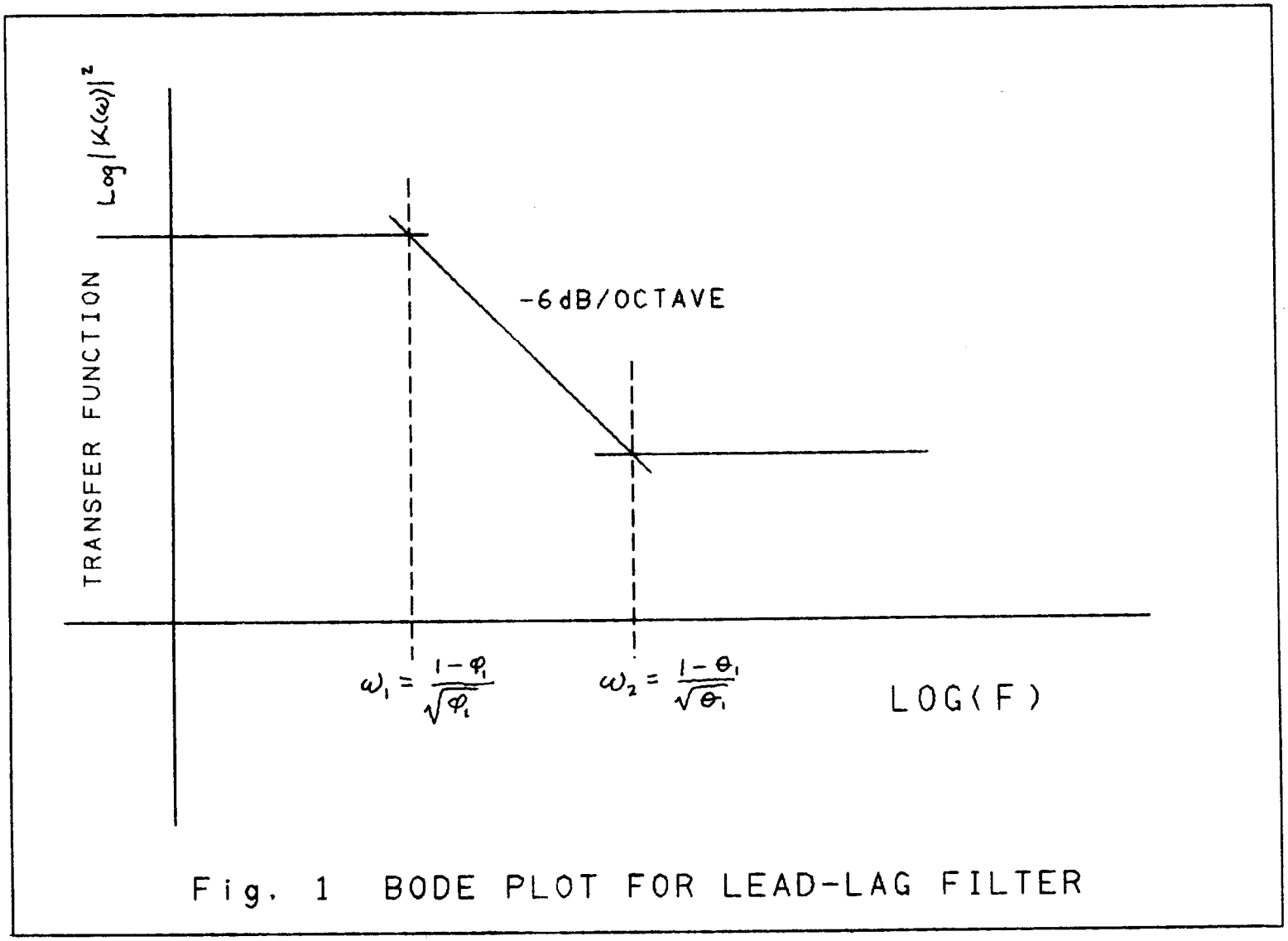
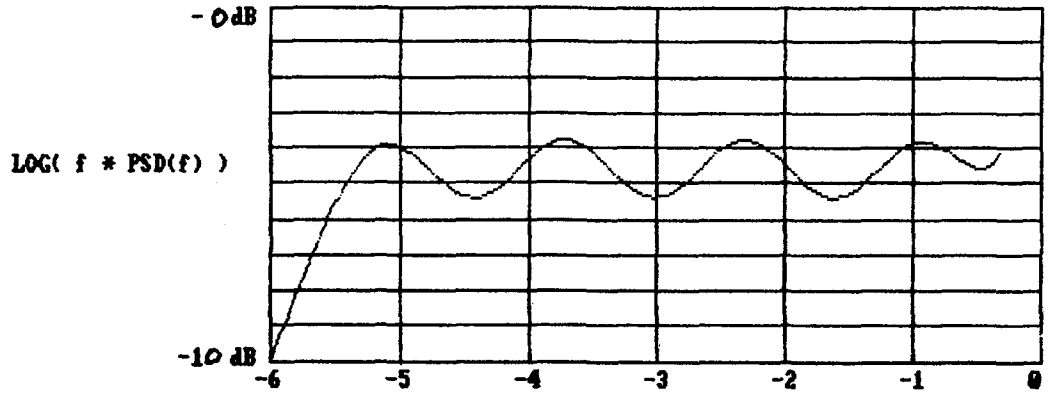
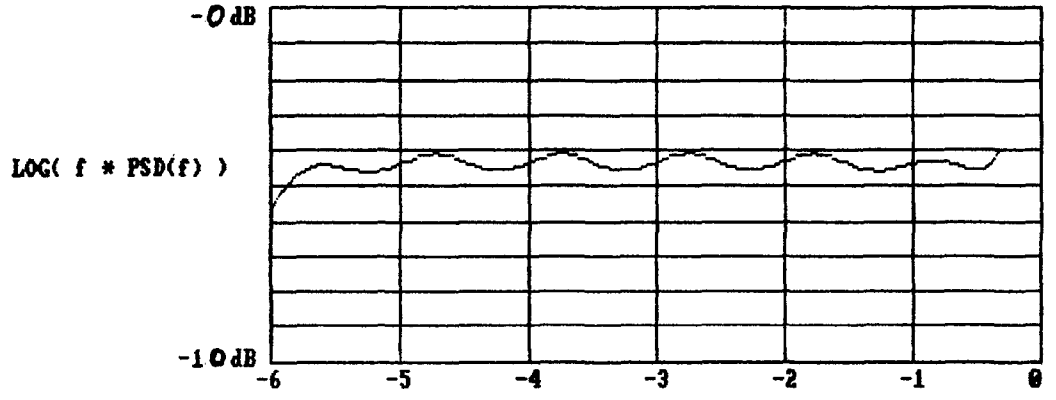


Fig. 1 BODE PLOT FOR LEAD-LAG FILTER



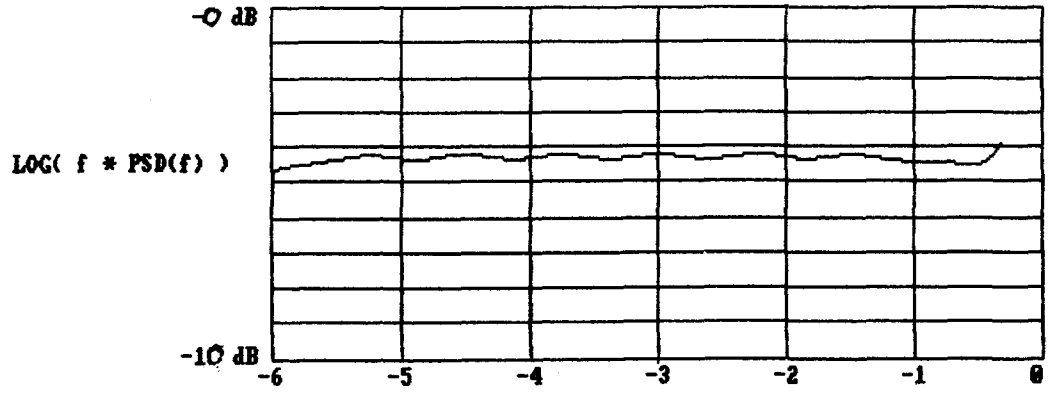
R = 5 FREQ. RATIO
M = 4 FILTER STAGES
PH(1) = .48 INITIAL PHI

FIGURE 2a FOUR-STAGE BARNES/JARVIS FILTER



R = 3.1 FREQ. RATIO
M = 6 FILTER STAGES
PH(1) = .38 INITIAL PHI

FIGURE 2b SIX-STAGE BARNES/JARVIS FILTER



R = 2.4 FREQ. RATIO
M = 8 FILTER STAGES
PH(1) = .31 INITIAL PHI

FIGURE 2c EIGHT-STAGE BARNES/JARVIS FILTER

Figure 2. Power Spectral Density Times Fourier Frequency

values can be stored, but the filter can recursively generate as many values as desired if all the values do not need to be retained in memory. A simple and reliable algorithm to convert uniformly distributed random numbers on (0,1) to normally distributed numbers with zero mean and unit variance is:

```
310        P=0
320        FOR K=1 TO 6
330             P=P+RND(1)-RND(1)
340        NEXT K
```

where each pass through this recursive routine produces a random, normal deviate.

To generate a million values would overflow the storage available in many computers. For the simulations reported here, every 100th value of $X(N)$ (the finite integral of $Y(M)$) was stored for the long τ-values. For the smaller τ-values only parts of the total one million data points were used. By always using the same seed for the random noise generator, one can extract various segments of the same noise sample without having all data points stored at one time.

The filter corresponding to lines 300 to 420 could, in principle, be replaced with an ARIMA(M,M-1) model by eliminating all the intermediate $Y(i)$ and $Y1(i)$ variables. Unfortunately, such a procedure requires many significant digits, and even double precision is totally inadequate. The form given here is adequate for most personal computers, but we recommend using double precision anyway.

Greenhall[4] has pointed out that the usual convention of starting out a filter with all past values set to zero causes systematic errors for all time. In effect, the flicker filter has significant auto-correlations for all lags in the data set and the initial zeros produce a non-representative sample. Greenhall went on to provide an initialization algorithm which circumvents these problems. Appendix B gives a computer program to calculate the initialization parameters.

Figure 3 shows the Allan variance obtained from the (finite) integral of a data set of a million flicker simulated numbers using the algorithms presented in this paper. That is, we simulated the phase of an oscillator perturbed by flicker noise modulating the oscillator frequency. Since the available computer could not handle a million points all at once, sample Allan variances were obtained by storing every one-hundredth phase data point, for calculating the Allan variances for the larger τ-values. For flicker FM, the Allan variance is constant[5], namely $f \times S(f) \times \log(4)$, where $S(f)$ is the spectral density of y (Figs. 2 and 5).

THE MANDELBROT GENERATOR

The Mandelbrot generator[1] sums a set of independent, band limited noises to approximate "Fractional Brownian Motions". This can be realized on a digital computer by filtering a white noise with a simple low pass filter:

$$Y_n = \phi_0 Y_{n-1} + P_n \tag{4}$$

The knee in the Bode-plot (Fig. 4) for the K-th low pass filter is located at:

$$(\omega_0, \frac{2\sigma^2}{(1-\phi_0)^2}) \tag{5}$$

That is, the low frequency asymptote of the filter transfer function has a power gain of $(1-\phi_0)^{-2}$ and the coefficient, ϕ, corresponding to the frequency at the knee is given by

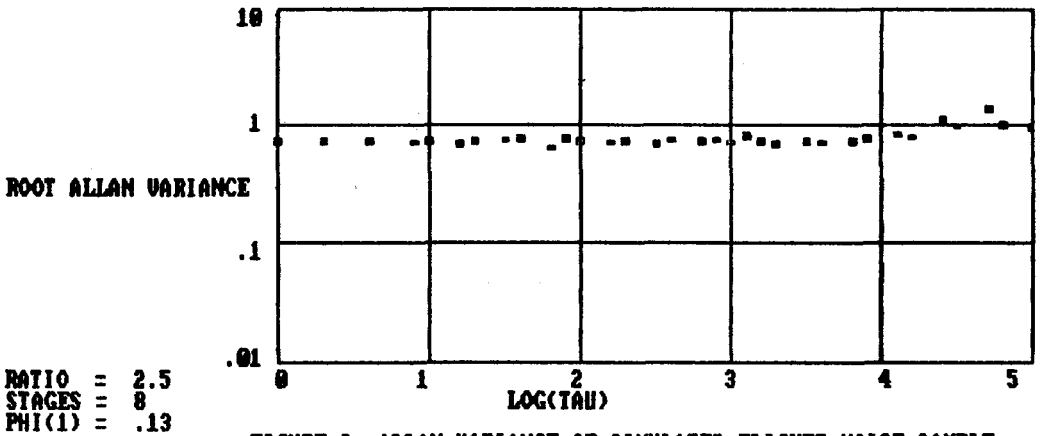


FIGURE 3 ALLAN VARIANCE OF SIMULATED FLICKER NOISE SAMPLE

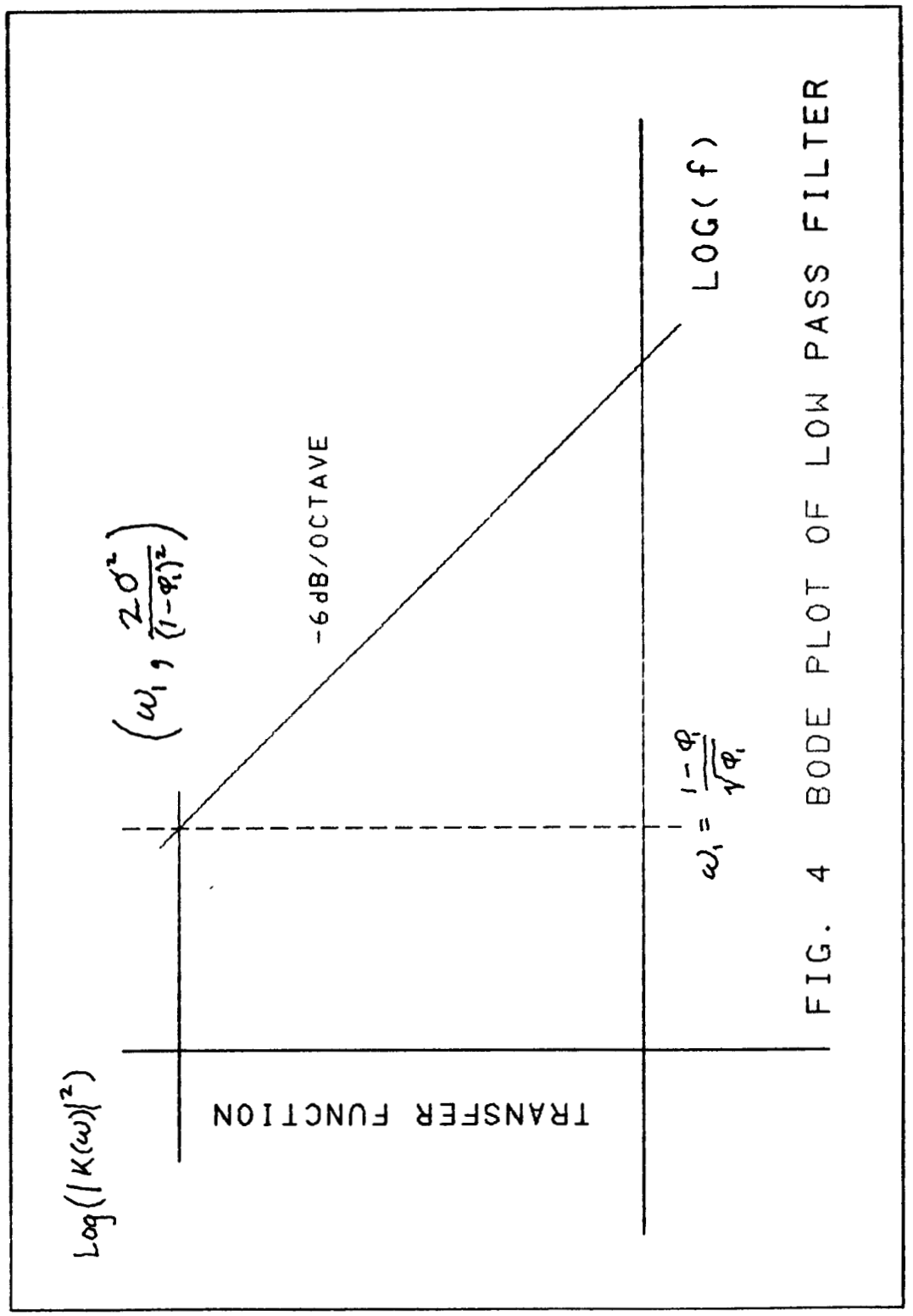


FIG. 4 BODE PLOT OF LOW PASS FILTER

Eq.(2), above (also see Appendix A). To simulate flicker noise, we want the knees of the various noises to fall along the curve $S = 1/f$.

As done in the treatment of the Barnes-Jarvis model, one first selects a frequency ratio, R, for successive noises, and then calculates the corresponding ϕ coefficient using Eq. (2) for each noise starting from the first frequency. The initial (highest) frequency is found by trial and error using the power spectral densities as calculated in Appendix A. Equating the low-frequency asymptote of each noise to $1/f$, one obtains the needed variance of the input white noise for each noise. The program used to calculate the ϕ's and amplitudes, $A(n)$, is shown below:

```
610     FOR N=1 TO M
620         W=C0/(R^N)
630         PH(N)=1#+.5#*W*(W-SQR(W*W+4#))
640         A(N)=SQR(.5#*(1#-PH(N))*SQR(PH(N)))
650         P=0:FOR J=1 TO 6:P=P+RND(1)-RND(1):NEXT J
660         Y(N)=P*A(N)/SQR(1#-PH(N)^2)
670     NEXT N
```

Lines 650 and 660 initialize each low pass filter to a random variable whose variance is the variance of its steady state output. This takes care of turn-on transients similar to the problems in the Barnes-Jarvis model. The coefficient C0 is chosen by observing the theoretical spectrum at the higher frequencies, near the Nyquist limit, 1/2T (see Fig. 5 and Appendix A).

The program which generates the flicker sample is shown below:

```
750     FOR N=1 TO NTOTAL
760         Z=0
770         FOR I=1 TO M
780             P=0:FOR J=1 TO 6:P=P+RND(1)-RND(1):NEXT J
790             Y(I)=Y(I)*PH(I)+A(I)*P
800             Z=Z+Y(I)
810         NEXT I
820     NEXT N
```

The output flicker noise is Z as determined recursively for each value of N. Figure 5 plots the theoretical power spectral density times f on a log-log plot similar to Fig. 2.

CONCLUSION

Large samples (a million or more values) of pseudo-random noise which approximates a flicker (or $1/f$) noise can be generated by rather simple recursive functions. The range and goodness of fit can be selected to meet any specific need and the methods are not compromised by the number of significant digits carried on most computers. The Barnes-Jarvis method and the Mandelbrot method seem to provide equally good results.

In regard to speed, however, the Barnes-Jarvis method was measured to be over 4 times faster than the Mandelbrot method running six-stage filters of comparable performance. The difference in speed is probably due to the fact that the Mandelbrot method needs one random number per stage per point, while the Barnes-Jarvis method requires only one random number per point regardless of the number of stages. The Barnes-Jarvis method ran in excess of 300 points per second using a compiled BASIC program in a micro-computer.

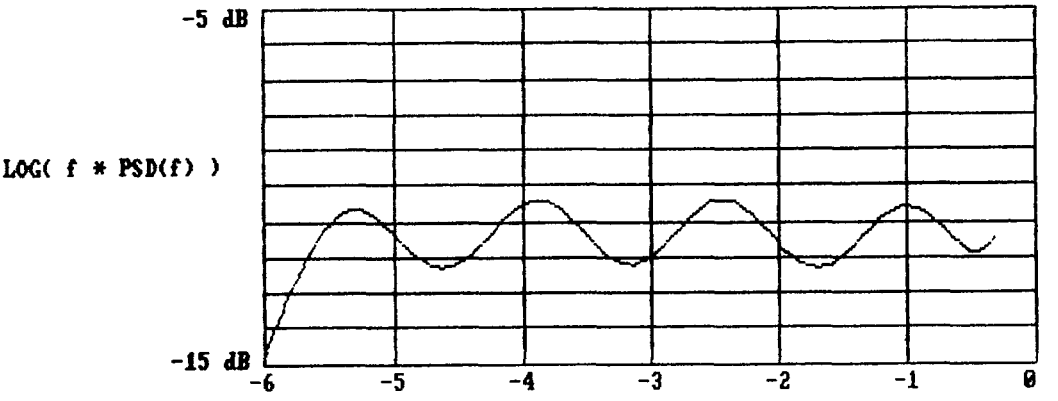


FIGURE 5a FOUR-STAGE MANDELBROT FILTER

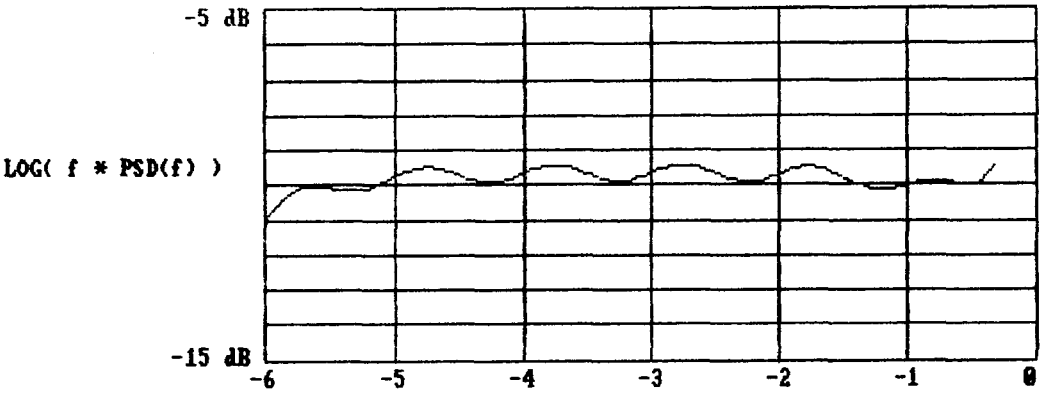


FIGURE 5b SIX-STAGE MANDELBROT FILTER

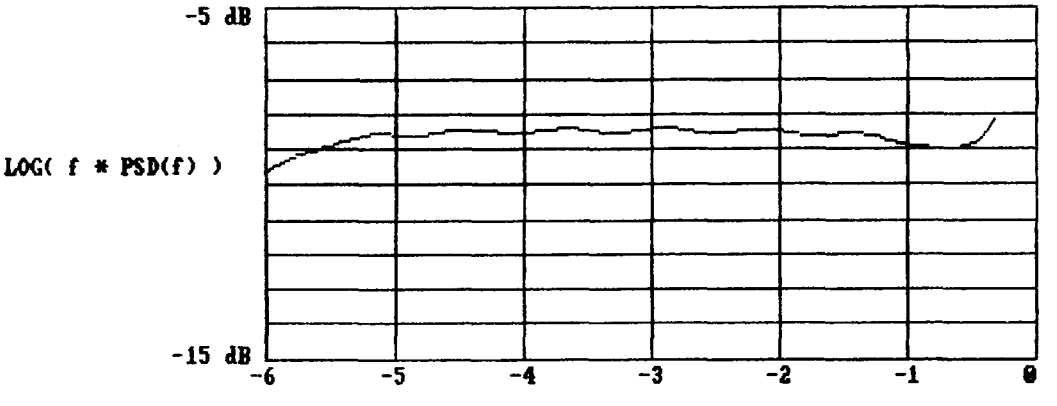


FIGURE 5C EIGHT-STAGE MANDELBROT FILTER

Figure 5. Power Spectral Density Times Fourier Frequency

Table 1. ALF(i,j)

RATIO = 2 PHI(1) = .3

.31449
.26035 .26333
.11022 .28084 .27924
.03183 .11429 .28500 .28480
.00826 .03280 .11459 .28632 .28629
.00208 .00850 .03278 .11468 .28668 .28667
.00052 .00214 .00849 .03278 .11471 .28676 .28676
.00013 .00054 .00214 .00848 .03278 .11472 .28679 .28679
.00003 .00013 .00054 .00214 .00848 .03278 .11472 .28679 .28679
.00001 .00003 .00013 .00054 .00214 .00848 .03278 .11472 .28679 .28679

RATIO = 2.5 PHI(1) = .325

.34366
.29249 .40792
.07255 .29740 .44051
.01254 .06936 .29817 .44641
.00203 .01188 .06871 .29833 .44737
.00033 .00192 .01175 .06861 .29836 .44752
.00005 .00031 .00190 .01173 .06859 .29837 .44755
.00001 .00005 .00030 .00190 .01173 .06859 .29837 .44755
.00000 .00001 .00005 .00030 .00190 .01172 .06859 .29837 .44755
.00000 .00000 .00001 .00005 .00030 .00190 .01172 .06859 .29837 .44755

RATIO = 3 PHI(1) = .35

.37363
.29450 .52478
.04720 .28356 .55965
.00548 .04290 .28232 .56381
.00061 .00495 .04239 .28218 .56428
.00007 .00055 .00489 .04233 .28217 .56433
.00001 .00006 .00055 .00488 .04233 .28217 .56433
.00000 .00001 .00006 .00055 .00488 .04233 .28217 .56433
.00000 .00000 .00001 .00006 .00054 .00488 .04233 .28217 .56433
.00000 .00000 .00000 .00001 .00006 .00054 .00488 .04233 .28217 .56433

RATIO = 3.5 PHI(1) = .375

.40452
.28370 .61773
.03162 .26287 .64943
.00265 .02792 .26111 .65214
.00022 .00233 .02760 .26097 .65237
.00002 .00019 .00231 .02757 .26096 .65238
.00000 .00002 .00019 .00230 .02757 .26095 .65239
.00000 .00000 .00002 .00019 .00230 .02757 .26095 .65239
.00000 .00000 .00000 .00002 .00019 .00230 .02757 .26095 .65239
.00000 .00000 .00000 .00000 .00002 .00019 .00230 .02757 .26095 .65239

RATIO = 4 PHI(1) = .4

.43644
.26820 .69282
.02195 .24240 .71998
.00140 .01907 .24073 .72173
.00009 .00121 .01888 .24062 .72184
.00001 .00008 .00120 .01887 .24062 .72185
.00000 .00000 .00008 .00120 .01887 .24062 .72185
.00000 .00000 .00000 .00007 .00120 .01887 .24062 .72185
.00000 .00000 .00000 .00000 .00007 .00120 .01887 .24062 .72185
.00000 .00000 .00000 .00000 .00000 .00007 .00120 .01887 .24062 .72185

RATIO = 4.5 PHI(1) = .425

.46951
.25167 .75512
.01575 .22407 .77784
.00079 .01358 .22266 .77899
.00004 .00068 .01347 .22259 .77905
.00000 .00003 .00067 .01347 .22259 .77905
.00000 .00000 .00003 .00067 .01347 .22259 .77905
.00000 .00000 .00000 .00003 .00067 .01347 .22259 .77905
.00000 .00000 .00000 .00000 .00003 .00067 .01347 .22259 .77905
.00000 .00000 .00000 .00000 .00000 .00003 .00067 .01347 .22259 .77906

RATIO = 5 PHI(1) = .45

.50390
.23570 .80843
.01164 .20820 .82727
.00047 .01002 .20707 .82804
.00002 .00040 .00996 .20702 .82807
.00000 .00002 .00040 .00995 .20702 .82807
.00000 .00000 .00002 .00040 .00995 .20702 .82807
.00000 .00000 .00000 .00002 .00040 .00995 .20702 .82807
.00000 .00000 .00000 .00000 .00002 .00040 .00995 .20702 .82807
.00000 .00000 .00000 .00000 .00000 .00002 .00040 .00995 .20702 .82809

RATIO = 6 PHI(1) = .5

.57735
.20764 .89840
.00685 .18306 .91130
.00019 .00594 .18235 .91166
.00001 .00017 .00591 .18234 .91167
.00000 .00000 .00017 .00591 .18233 .91167
.00000 .00000 .00000 .00017 .00591 .18233 .91167
.00000 .00000 .00000 .00000 .00017 .00591 .18233 .91167
.00000 .00000 .00000 .00000 .00000 .00017 .00591 .18234 .91168
.00000 .00000 .00000 .00000 .00000 .00000 .00016 .00591 .18230 .91121

APPENDIX A

An "exponential" filter or simple low pass filter has the following form:

$$Y_n = \phi Y_{n-1} + a_n \tag{A1}$$

where $0 < \phi < 1$, and a_n are random normal deviates with zero mean and variance σ^2. An equivalent representation[4] of this filter in terms of the impulse response function is:

$$Y_n = a_n + \phi a_{n-1} + \phi^2 a_{n-1} + \ldots \tag{A2}$$

Taking the expectation value of the square of (A2) one obtains:

$$\begin{aligned} E[Y_n^2] &= \sigma^2 \sum_{i=0}^{\infty} \phi^{2i}, \\ &= \frac{\sigma_a^2}{1-\phi^2} \end{aligned} \tag{A3}$$

since the a_n are independent. The initial conditions for such a low pass filter (e.g., as used in the Mandelbrot method), can be taken as a random normal deviate with this variance. This is the source of lines 650 and 660 in the text.

Box and Jenkins [3] also show that the PSD of $Y(n)$ is given by:

$$S(f) = \frac{2\sigma_a^2}{1 + \phi^2 - 2\phi\cos(\omega)} \tag{A4}$$

At $f = 0$, the value $S(0)$ is just $2\sigma_a^2/[(1-\phi_0)^2]$ and we define the cutoff frequency as that frequency for which $S(f) = S(0)/2$. Using the first two terms in the Taylor series expansion for $\cos(2\pi f)$ in (A4), one obtains:

$$\omega_0 = (1 - \phi_0)/\sqrt{\phi_0} \tag{A5}$$

or, equivalently:

$$\phi_0 = 1 + \frac{\omega_0}{2}(\omega_0 - \sqrt{\omega_0^2 + 4}) \tag{A6}$$

Also, (see lines 260, 280, and 630):

$$\begin{aligned} \frac{2\sigma_a^2}{(1-\phi_0)^2} &= \frac{1}{\omega_0} = \frac{\sqrt{\phi_0}}{1-\phi_0} \\ \sigma_a^2 &= \frac{1}{2}\sqrt{\phi_0}(1-\phi_0) \end{aligned} \tag{A7}$$

The power spectral density (PSD) of the Barnes-Jarvis filter at (angular) frequency W can be calculated using the following routine:

```
500     S=2#
510     FOR J=1 TO M
520         S=S*(1#+TH(J)^2-2#*TH(J)*COS(W))
530         S=S/(1#+PH(J)^2-2#*PH(J)*COS(W))
540     NEXT J
```

and for the Mandelbrot method the PSD can be calculated using:

```
800     S=0#
810     FOR J=1 TO M
820         S=S+2#*(A(J)^2)/(1#+PH(J)^2-2#*PH(J)*COS(W))
830     NEXT J
```

where S is the PSD. S is multiplied by $W/2\pi$ for Figures 2 and 5 to display the errors from a perfect $1/f$ noise. The PH(J)'s and TH(J)'s are those calculated for the appropriate noises.

APPENDIX B
Initialization of a Barnes-Jarvis Filter

Following Greenhall[4], the covariance program (in BASIC) can be written in the form:

```
1210        ' COVARIANCE PROGRAM
1220        FOR N=1 TO M
1230            C(1,N)=1#:D(1,N)=1#
1240            FOR I=2 TO M
1250                I1=I-1:F=ALF(I1)-BET(N)
1260                IF I<>N THEN F=F/(BET(I1)-BET(N))
1270                C(I,N)=C(I1,N)*F
1280                F=ALF(I1)+BET(N)-ALF(I1)*BET(N)
1290                F=F/(BET(I1)+BET(N)-BET(I1)*BET(N))
1300                D(I,N)=D(I1,N)/F
1310            NEXT I
1320        NEXT N
1330        '
1340        FOR I=1 TO M
1350            FOR J=1 TO M
1360                SUM=0#
1370                FOR N=1 TO I
1380                    F=C(I,N)*D(J,N)/(BET(J)+BET(N)-BET(J)*BET(N))
1390                    IF I<>N THEN F=F/(BET(I)-BET(N))
1400                    SUM=SUM+F
1410                NEXT N
1420                RZ(I,J)=SUM*(ALF(I)-BET(I))*(ALF(J)-BET(J))
1430            NEXT J
1440        NEXT I
```

where BET(I) = 1 - PH(I) and ALF(I) = 1 - TH(I). The desired covariance matrix is RZ(I,J).

The Choleski square root of the covariance matrix can be computed with the following BASIC program:

```
1450        ' CHOLESKI SQUARE ROOT
1460        FOR I=1 TO M
1470            G=RZ(I,I)
1480            FOR K=1 TO I-1
1490                G=G-ALZ(I,K)^2
1500            NEXT K
1510            ALZ(I,I)=SQR(G)
1520            FOR J=I+1 TO M
1530                TEMP=RZ(J,I)
1540                FOR K=1 TO I-1
1550                    TEMP=TEMP-ALZ(I,K)*ALZ(J,K)
1560                NEXT K
1570                ALZ(J,I)=TEMP/ALZ(I,I)
1560            NEXT J
1570        NEXT I
```

The lower triangular matrix, ALZ(J,I), contains the coefficients needed to initialize the Y1(I)'s. Table B1 lists the coefficients for some ratios and initial PHI(1) values. For fewer than 10 stages the matrix can simply be truncated.

The next program initializes the Barnes-Jarvis filter:

```
1600        ' INITIALIZATION
1610        FOR I=0 TO M
1620            P=0:FOR J=1 TO 6:P=P+RND(1)-RND(1):NEXT J
1630            U(I)=P
1640        NEXT I
1650        Y1(0)=U(0)
1660        '
1670        FOR I=1 TO M
1680            Z(I)=0
1690            FOR J=1 TO I
1700                Z(I)=ALZ(I,J)*U(J)+Z(I)
1710            NEXT J
1720            Y1(I)=Y1(I-1)+Z(I)
1730        NEXT I
```

The components Y1(1),..., Y1(M) are used in the first step of the filter (Lines 300 - 410).

REFERENCES

[1] B. Mandelbrot, "A Fast Fractional Gaussian Noise Generator", Water Resources Research, Vol. 7, No. 3, June 1971.

[2] J. Barnes and S. Jarvis, "Efficient Numerical and Analog Modeling of Flicker Noise Processes", National Bureau of Standards Technical Note 604, June 1971.

[3] G.P.E. Box and G.M. Jenkins, "Time Series Analysis", Holden-Day, San Francisco, CA. 1970.

[4] C. Greenhall, "Initializing a Flicker-Noise Generator", IEEE Trans. on Instrumentation and Measurement, Vol. IM-35 No. 2, June 1986.

[5] D. Allan, "Statistics of Atomic Frequency Standards", Proc. IEEE, Vol. 54, pp 221-230, February 1966.

The work of the second author was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.